METHODS AND STRUCTURES FOR AN EXTENSIBLE RAID STORAGE ARCHITECTURE

Background of the Invention

1. Field of the Invention

The invention relates to storage subsystem architectures and in particular to a RAID storage subsystem architecture that applies SAN principles and technology to the internal architecture of the storage subsystem.

2. Discussion of Related Art

Computing storage subsystems are evolving at a rapid pace to require, at once, high capacity, high performance and high reliability. Disk drive technology has evolved to enable large capacities in individual disk drives. As applied in storage subsystems with multiple drives to achieve higher total storage capacity, each high capacity disk drive gives rise to performance bottlenecks as well as significant reliability problems. Where for example an entire request to store or retrieve data is directed to a single disk drive, the throughput of the storage system will be that of the single disk drive and the reliability of the subsystem will be that of a particular disk drive.

Redundant arrays of inexpensive disks ("RAID") storage systems have addressed these needs by providing redundancy for reliability and management techniques to achieve higher performance. Specifically, RAID subsystems apply various management techniques (often referred to as RAID "levels") to provide redundancy in the storage of data on the disk drives such that failure of a single disk drive does render the entire subsystem unusable. Other RAID techniques ("striping") distribute the data over multiple disk drives to achieve the benefit of multiple disk drives processing a single larger I/O request to read or write data. Where *N* disk drives are used to process a single I/O request, the time to complete the request as compared to a single drive is on the order of 1/*N*.

The 'array" of multiple disk drives in a RAID storage subsystem is managed by a RAID storage controller device. The storage controller typically includes a general purpose microprocessor with associated program memory,

cache memory for caching data sent to and from the disk drive array, "back-end" interfaces to adapt the controller to the disk drive array (i.e., SCSI and/or Fibre Channel interface controllers), a "front-end" interface to couple the controller to one or more host systems, etc. The storage controller manages the disk array to make the array appear to a host computer as a large single disk drive that offers improved performance and reliability as compared that of a single disk drive.

To further enhance reliability and performance, RIAD subsystems also are known to utilize multiple such storage controllers. The multiple storage controller are often configured and managed to provide redundancy such that failure of a single storage controller does not render the subsystem inaccessible. The multiple controllers may also be configured to enhance performance of the storage subsystem by providing parallel processing by multiple controllers of multiple host system I/O requests. The load of I/O requests may therefore be distributed over the plurality of storage controllers to reduce the total processing time required for a series of I/O requests that may be processed in parallel.

Such multiple controller architectures still suffer from certain performance bottlenecks. For example, it is common that the multiple controllers share a common connection to the disk drives in the disk array. Shared use of the common disk interface can therefore become a performance restriction for multiple controllers in processing multiple I/O requests in parallel. Similarly, the number of I/O connections ("channels") for connecting the multiple controllers to host systems may be a bottleneck.

Addition of disk drives without corresponding addition of communication channels and associated back-end control functionality could easily saturate existing disk channels. However, presently known architectures do not readily lend themselves to addition of disk drive communication channels independent of controllers having integrated front-end and back-end control functions. Present architectures generally require that the maximum anticipated bandwidth requirements of the back-end communication channels be anticipated in the original design and architecture of the storage subsystem. When applied to lower-end applications requiring only a portion of such capacity, the subsystem is

"over designed" in that excess bandwidth capacity is unused and therefore wasted and costly.

Some prior architectures called for "N-way" connectivity among the controllers and the disk drives. In other words, any number "N" of controllers shared access to a common set of disk drives via a common, single communication channel. However, such architectures can rapidly saturate the single, shared communication channel when additional disk drives are added to increase storage capacity. Even where multiple communication channels are utilized, the architecture calls for each controller to access each disk drive adding cost and complexity to each of the N controllers.

In general, present high performance RAID storage subsystems suffer from lack of flexibility in configuring the multiple controllers and multiple disk storage devices or modules. It is therefore desirable to improve the flexibility of such configurations to permit easier enhancement of performance and reliability characteristics of a storage subsystem.

Summary of the Invention

The present invention solves the above and other problems, thereby advancing the state of the useful arts, by providing a storage subsystem architecture that divides the controller function between front-end controller and back-end controller and that applies storage area network ("SAN") techniques and devices within the storage subsystem to interconnect the front-end controllers and back-end controllers. SAN components are known and applied outside the storage subsystem for interconnection of such storage subsystems to host computers and other computing subsystems. In the context of this invention, SAN switches are applied ***within*** the storage subsystem to permit more flexible configuration of front-end and back-end control devices within the storage subsystem.

A plurality of back-end storage controllers and a plurality of front-end controllers are configured within a storage subsystem interconnected by a SAN switching network that permits broad flexibility in interconnecting the various

controllers. The front-end controllers ("FECs") are dedicated to "front-end" interfacing to host computer systems and are devoid of circuits and functions to control the disk array devices. The back-end controllers ("BECs") are dedicated to "back-end" control of the disk arrays and are devoid of circuits and functions to interface directly with the attached host systems. In this architecture, the FECs and BECs are simpler than prior integral controllers that provided both front-end and back-end control functions.

Each FEC and BEC includes a SAN interface to connect to the SAN switches. The SAN switches therefore provide flexible interconnection between virtually any number of front-end controller and any number of back-end controllers. Such a storage subsystem may thereby be flexibly configured to add additional back-end control where required for back-end performance or reliability enhancement and may be configured to add additional front-end controller when required for front-end performance and reliability.

By providing such configuration flexibility and simpler FEC and BEC devices that segregate their respective functions, the storage subsystem is more scalable than prior known architectures. Additional FECs may be added to alleviate host communication bottlenecks independent of BEC control functions. Conversely, BECs may be added to alleviate disk communication bottlenecks independent of FEC control functions.

Brief Description of the Drawings

Figure 1 is a block diagram of a RAID storage subsystem as presently known in the art.

Figure 2 is a block diagram of an exemplary RAID storage subsystem in accordance with the present invention.

Figure 3 is a block diagram of a front-end controller of figure 2.

Figure 4 is a block diagram of a back-end controller of figure 2.

Detailed Description of the Preferred Embodiments

While the present invention is susceptible to various modifications and alternative forms, specific embodiments thereof have been shown by way of example in the drawings and will herein be described in detail. It should be understood, however, that it is not intended to limit the invention to the particular form disclosed, but on the contrary, the invention is to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the invention as defined by the appended claims.

Figure 1 is a block diagram of a typical multi-controller RAID storage subsystem 1 as presently practiced in the art. A plurality of storage controllers 100 and 110 (Redundant Dual Access Controllers ("RDACs") #1 and #2) within the subsystem provide both front-end interfacing to hosts 170..174 via medium 160 and back-end interfacing to a pair of storage modules 120 and 130 via medium 150. The storage modules 120 and 130 each include a plurality of disk drives 122 and 132, respectively. Each storage controller 100 and 110 is coupled to medium 160 via a front-end interface element 102 and 112, respectively. Storage controller 120 is coupled to both storage modules 120 and 130 via back-end interfaces 104 and 106, respectively. Storage controller 110 is coupled to both storage modules 120 and 130 via back-end interfaces 114 and 116, respectively and through communication media 150.

As is known in the art, the host communication media 160 may be any of several well-known media including: parallel SCSI, Fibre Channel, Ethernet (or other local area network media), etc. Similarly, it is known in the art that the back-end communication media 150 may be any of several well-known media including parallel SCSI, Fibre Channel, ATA, EIDE, etc. Those skilled in the art will recognize that depending upon the choice of media elements 150 and 160 may include appropriate switches, hubs and other connectivity devices as required for the particular communication medium.

This exemplary known architecture provides redundant connectivity within the storage subsystem between the storage controllers and the storage modules. As noted above, this known architecture is inflexible in terms of scalability in that

the front-end control functions (i.e., performed within 102) are integrated on a single controller along with the back-end control functions (i.e., performed within 104 and 106). If the subsystem has a need for enhancing back-end performance, additional back-end performance in the form of back-end interface elements and functions are coupled with front-end control circuits and functions. Likewise, if additional front-end processing power for host generated I/O requests is required, the additional controllers are integrated with, potentially extraneous back-end control circuits and functions. Furthermore, the interconnection of additional controllers with existing storage modules may be cumbersome depending on the type of connections used.

More specifically, the front-end controllers perform processing related to transactions with attached host computer systems and higher level storage management functions while back-end controllers perform processing related to RAID management of the storage devices and lower level controls within the storage subsystem. Each controller therefore addresses different aspects of the overall performance of the storage subsystem. Both front-end and back-end controllers confront problems with available capacity to handle host I/O transactions. The size and frequency of host I/O requests impacts the performance requirements of both the front-end controllers and the back-end controllers. Back-end controllers confront problems relating to interfacing with disk drives and the associated communication channels used therefore. In particular, the back-end controller is matched to a communication channel bandwidth associated with a number disk drives. The configuration of back-end controllers is therefore preferably matched to the performance characteristics of the disk drive attached to it and the associated communication channel bandwidth. A few high performance disk drives can saturate the communication channels used to communicate with back-end controllers. Additional communication bandwidth for disk drives may therefore require additional back-end controllers to accommodate the potential saturation of the disk interface channel. The needs to scale the front-end transaction processing performance and high end storage management is largely distinct from the needs to scale the back-end performance for RAID management and lower level storage management functions. Though not enabled by prior

techniques, it is useful to isolate these functions to permit independent scaling of the performance of front-end control functions and scaling of the back-end control functions.

Figure 2 is a block diagram showing the architecture of a storage system 2 in accordance with the present invention wherein the front-end control circuits and functions are separated from the back-end control circuits and functions. As used herein, "front-end" refers principally to the host system interfacing functions. Exemplary of the functions performed by such front-end controllers are higher level I/O request processing such as RAID storage management for redundancy, RAID logical to physical storage mapping, hierarchical storage management, network file protocol support, high level data striping, backup and restore, routing of I/O requests among controllers, and management functions to map storage to data applications. As used herein, "back-end" refers to lower level control functions relating to disk drive interfacing and associated physical I/O operations on the disk drives. Exemplary of such back-end control functions are high availability storage functions (i.e., RAID management), high performance disk interfacing, high bandwidth I/O management, local device management and data management primitives such as data snapshots and data migration.

Caching of data may occur in both front-end and back-end controllers -- typically for different purposes and for enhancing performance of different aspects of the storage subsystem. Those skilled in the art will recognize that the definitions herein of high level or front-end functions as compared to lower level of back-end functions are matters of design choice. Other definitions and divisions of functions among the controllers are possible and within the scope of the present invention. Key to the invention is some division of functions between a front-end controller and a back-end controller allowing independent scaling of the controllers.

The two layers (front-end and back-end) communicate via a SAN architecture layer preferably using an intelligent, structured interface protocol. The interface protocol may utilize a custom design protocol because this architecture is internal to the storage subsystem and need not be exposed external to the subsystem. In the alternative, the structured interface protocol may apply industry

standards such as I^2O or the Intel Virtual Interface Architecture. Again, such interface protocols and structures issues constitute well known design choices for those skilled in the art.

In particular, storage system 2 includes a plurality of front-end control elements 200, 208 and 216. Each front-end controller includes one or more front-end interface elements (202, 210 and 218, respectively) to connect the front-end control element (FEC) to one or more host systems 170..174 via a host communication media 160..161. FECs may be connected to a plurality of host system communication media as required for flexible connectivity to host systems. For example, media 160 and 161 may be separate segments of a common communication media type or may even be different types. As noted above, the communication medium used between FECs and host systems may be any of several well-known types as discussed above.

Each FEC (200, 208, 216) also includes one or more intra-subsystem SAN interfaces (204 and 206, 212 and 214, and 220 and 222, respectively). Intra-subsystem SAN interfaces 204, 206, 212, 214, 220 and 222 are referred to as "intra-subsystem" to distinguish from SAN interfaces that may be used in a storage subsystem to connect to SAN components external to the storage subsystem. Such external SAN interfaces are not relevant to the operation and structure of the present invention. As used herein below "SAN interface" refers to intra-subsystem SAN interfaces as distinct from any SAN interfaces that may be present on a controller for interfacing external to the storage subsystem.

Each FEC includes one or more SAN interface elements connecting the FEC to the SAN switches 250 and 252 via SAN communication media 254. There are preferably at least two SAN switches 250 and 252 to permit such redundant connectivity from the front-end control elements to the plurality of back-end control elements discussed below. There may be any number of such redundant links but in the preferred embodiment, two links from each front-end control element, one to each of two SAN switches, is considered necessary and sufficient. Where reliability of the front-end control communication with the back-end control elements is

deemed less important, a single connection between a front-end control element and the SAN switches may be adequate.

Storage system 200 also includes a plurality of back-end control elements 260, 264, 268 and 272 preferably configured as shown in redundant pairs (260 and 264 as a first pair and 268 and 272 as a second pair). Each back-end control element (BEC) includes a SAN interface element (262, 266, 270 and 274, respectively). Each BEC of a redundant pair is connected to one of the two redundant SAN switches 250 and 252 via SAN communication media 256. Specifically as exemplified in figure 2, back-end control element 260 (BEC) connects to SAN switch 250 via SAN interface 262. BEC 264 connects to SAN switch 252 via SAN interface 266. BEC 268 connects to SAN switch 252 via SAN interface 270 and lastly, BEC 272 connects to SAN switch 250 via SAN interface 274.

Each BEC connects to a storage module 280 or 290 comprised of a plurality of disk drives 282 and 292, respectively. Each BEC of a redundant pair preferably connects to one of the storage modules. For example, as shown in figure 2, BEC 260 connects to storage module 280 via media 150 and BEC 264, the other BEC of the redundant pair of 260 and 264, also connects to storage module 280 via media 150. It is also possible for each BEC to provide a pair of redundant links to its associated storage module. For example, as shown in figure 2, redundant pair of BECs 268 and 272 are each redundantly connected to storage module 290 via a redundant pair of communication links in media 150. As noted above, communication media 150 between the BECs and the storage modules may be any over several well-known types as discussed above.

Those skilled in the art will recognize that the specific configuration (topology) shown in figure 2 is intended merely as exemplary of one possible embodiment in accordance with the present invention. The present invention provides for the segregation of front-end control functions and back-end control functions into distinct circuits with a SAN architecture interconnecting the elements. A wide variety of alternate configurations and topologies will be recognized by those skilled in the art. Further, the number of FECs, BECs and

SAN switches and the grouping of those devices into pairs, is intended merely as exemplary of one preferred embodiment. Any number of FECs, BECs and SAN switches may be configured in a system in accordance with the present invention. In a particular application, the number of such controllers and SAN switches is determined by matching the bandwidth and transaction processing capability of the components with the subsystem requirements for that application. The individual modules and components (FECs, BECs, disk drives, SAN switches, etc.) of the storage subsystem in accordance with the present invention may be dynamically reconfigured by a user to modify performance characteristics to fit changing demands on the storage subsystem.

The present invention expresses the preference for at least pairs of SAN switches and pairs of BECs to ensure redundancy throughout the connections from a host system through to the individual disk drive devices. Any number of FECs, BECs and SAN switches, paired or not, may be configured within the intended scope of the present inventions.

As noted above, the SAN switches (250 and 252) and associated SAN communication media 254 and 256 may apply any of several existing SAN architectures. The SAN switches and associated communication media may be any that allows the passing of data and I/O requests between the FECs and the BECs with low latency (i.e., less than 10 microseconds). Typical of such devices/media are PCI buses, local area network (LAN) connections (i.e., Ethernet or Gigabit Ethernet, etc.), Fibre Channel SAN switch devices and media, InfiniBand (www.infinibandta.org) and ServerNet (developed by Tandem and presently sold by Compaq). The ideal configuration involves a switch that allows for bandwidths that scale with the number of devices (FECs and BECs) that are added to the SAN. Present market forces and technology factors suggest that InfiniBand is a preferred embodiment of the SAN communication media.

Figure 3 is a block diagram of a typical FEC device in accordance with the present invention. As noted above, the FEC of the present invention is similar to a storage controller as presently known in the art and as shown in figure 1 except

that it is devoid of back-end control functions and circuits. Rather, the FEC has a redundant SAN interface to permit flexible connectivity to back-end control elements through the SAN layer.

In particular, FEC 200 is shown in additional detail in figure 2. FEC 200 is intended as exemplary of all FECs shown in figure 2 above. In the preferred embodiment, FECs are not identical devices. As noted above, each FEC may provide a different type of host (front-end) interface to permit added flexibility in the connectivity of the storage system. The different types of host interfaces may include different physical interfaces and protocols or merely different logical interfaces provided on a common physical medium. In addition, different front-end interfaces may provide varying functions for particular connection application. For example, network file protocols may be directly supported in a particular FEC while another FEC may provide only lower level block level access interface functions.

In the preferred embodiment. FEC 200 includes a general purpose CPU 300 that controls overall operation of the FEC and processes I/O requests received from the front-end interface element 202 and received from the back-end devices connected through SAN interfaces 204 and 206. Programmed instructions and data for operation of CPU 300 are stored in program memory 304. Data sent to or from the host systems or the BECs is cached in cache memory 302 to improve controller performance. DMA 306 assists CPU 300 in transferring data among the various components. All components communicate via processor bus 350.

Those skilled in the art will recognize that the block diagram of figure 3 is intended merely as exemplary or suggestive of the design of an FEC. The specific compliment of components associated with the CPU and the specific bus or buses that interconnect those components is a matter of design choice well-known to those skilled in the art. For example, a front-end controller may optionally include RAID parity assist (RPA) computation devices for other higher level RAID management support in the FEC. Key to the FEC design shown in figure 3 is that the FEC is devoid of back-end disk drive interface components.

Rather, that function is segregated onto a back-end controller element. The FEC therefore preferably performs necessary mapping of logical storage addresses (supplied by host I/O requests) into physical storage locations conveyed to appropriate BECs to perform the host requested I/O operation. The SAN interfaces 204 and 206 permit flexible interconnection of the FEC with a number of BEC elements via the SAN intermediate layer. Further, as noted above, intelligent I/O interfacing protocols and APIs are preferably implemented within the FEC and BEC to permit a structured, standardized interface between the layers through the SAN switch intermediate communication layer.

Figure 4 is a block diagram of an exemplary back-end control element 260 (BEC) in accordance with the present invention. BEC 260 is representative of all BEC elements shown above in figure 2. BECs are preferably identical in design, though as noted they may vary in accordance with specific needs of a particular storage system application. For example, different BECs within a storage system may each provide a different back-end interface medium to connect to a set of disk drives. A first BEC in a storage system may use parallel SCSI for example to connect to a storage module while a second BEC in the same storage system may use Fibre Channel to connect to a storage module. Similarly, a first BEC may provide a single connection to a storage module while another BEC in the same system may provide redundant links to another or the same storage module. Or, for example, groups of BECs may be tuned for different performance characteristics. Some BECs may be tuned to high bandwidth back-end performance requirements while others could be tuned high I/O transaction rate requirements. Still other BECs may be tuned for tape storage as distinct from disk storage. Such flexible configurations are useful in hierarchical storage management applications where a multitude of storage media are incorporated into a single storage subsystem each medium having different access and performance characteristics.

As above, the BEC performs the lower level functions of interfacing with the disk drives directly. Lower level physical I/O operations are performed by the BEC. As noted above, a key to the architecture of the BEC of the present

invention is that it is devoid of functions and associated circuits for performing host interfacing (front-end interfacing). Otherwise, BECs are substantially similar to the general structure of FECs. Programmed instructions and data for operation of CPU 300 are stored in program memory 304. Data sent to or from the disk drives or the FECs is cached in cache memory 302 to improve controller performance. DMA 306 assists CPU 300 in transferring data among the various components. As noted above, RAID storage management functions are preferably performed within BEC 260. RPA 308 (RAID Parity Assist) aids CPU 300 in rapidly computing parity values for RAID storage management functions within the BEC. All components communicate via processor bus 450.

In particular, BEC 260 includes one or more SAN interfaces 262 to connect to the SAN communication media 256. The SAN interfaces 262 are coupled via bus 450 to disk interfaces 400 and 402 which, in turn, coupled via bus 150 to storage modules and/or individual disk drives. As shown in figure 4, disk interfaces 400 and 402 include all intelligence required to interface with a front-end control element via bus 450 and SAN interface 262. Those skilled in the art will recognize that in particular applications it may be beneficial to implement the FEC and BEC as identical hardware components each implementing its particular designated function. Such identity of the hardware components permits more flexible replacement of spare parts in the subsystem. Further, those skilled in the art will recognize that many of the components in an FEC or BEC may be integrated into higher level integrated circuits incorporating many discrete functions into a VLSI custom circuit. Such design choices are well-known to those skilled in the art. Key to the BEC of the present invention is that it is devoid of front-end functions and associated circuits. Rather, it performs only the back-end functions of low level disk drive command processing. Interfacing with higher level front-end control elements is provided via the SAN interfaces of the BEC.

The present invention as described above provides a key benefit in that the architecture can be flexibly scaled to different bandwidth requirements unique to particular applications. As back-end performance becomes a bottleneck, additional BECs may be easily integrated. Likewise, as front-end I/O processing

performance becomes a bottleneck in system throughput, additional FECs may be added to improve I/O processing performance. Further, as new or additional host interface channels or protocols are required, additional FECs having different host channel interfaces and/or protocols may be added. The architecture of the present invention therefore improves flexibility and scalability of the storage subsystem to allow customization and adaptation to particular needs of particular applications.

While the invention has been illustrated and described in detail in the drawings and foregoing description, such illustration and description is to be considered as exemplary and not restrictive in character, it being understood that only the preferred embodiment and minor variants thereof have been shown and described and that all changes and modifications that come within the spirit of the invention are desired to be protected.